

TrialName

epool@localhost






















Data Dictionary

18/04/2019

Table of contents

epool@localhost	4
1. Tables	4
1.1. Table: azienda	4
1.2. Table: foto	5
1.3. Table: passaggio	6
1.4. Table: pdf	7
1.5. Table: prenotazione	8
1.6. Table: privata	9
1.7. Table: pubblica	10
1.8. Table: segnalazione	11
1.9. Table: societa	12
1.10. Table: sosta	13
1.11. Table: tappa	14
1.12. Table: tragitto	15
1.13. Table: tragitto_prenotazione	16
1.14. Table: utente	17
1.15. Table: utente_aziendale	18
1.16. Table: utente_premium	19
1.17. Table: utente_semplice	20
1.18. Table: valutazione	21
1.19. Table: veicolo	22
2. Views	23
2.1. View: media_voto_utente	23
2.2. View: prenotazione_capienza	24
2.3. View: veicoli_disponibili	25
3. Procedures	26
3.1. Procedure: getFoto	26
3.2. Procedure: getPDF	27
3.3. Procedure: inserisciPassaggio	28
3.4. Procedure: InserisciPrenotazione	29
3.5. Procedure: InserisciPrenotazioneAziendale	30
3.6. Procedure: InserisciSegnalazione	31
3.7. Procedure: InserisciTappa	32
3.8. Procedure: inserisciTragitto	33
3.9. Procedure: InserisciValutazione	34
3.10. Procedure: InsertFoto	35
3.11. Procedure: InsertPdf	36
3.12. Procedure: Login	37
3.13. Procedure: LoginAziendale	38
3.14. Procedure: LoginType	39
3.15. Procedure: PrenotazioneP	40
3.16. Procedure: printf	41
3.17. Procedure: RegistrazioneAziendale	42
3.18. Procedure: RegistrazioneUtente	43

Legend






-  Primary key
-  Primary key disabled
-  User-defined primary key
-  Unique key
-  Unique key disabled
-  User-defined unique key
-  Active trigger
-  Disabled trigger
-  Many to one relation
-  User-defined many to one relation
-  One to many relation
-  User-defined one to many relation
-  One to one relation
-  User-defined one to one relation
-  Input
-  Output
-  Input/Output
-  Uses dependency
-  User-defined uses dependency
-  Used by dependency
-  User-defined used by dependency

epool@localhost


1. Tables

1.1. Table: azienda

Columns





		Name	Data type	Description / Attributes
		NOME	varchar(30)	
		INDIRIZZO	varchar(50)	
		TELEFONO	int(10, 0)	
		RECAPITO	int(10, 0)	

Unique keys


Columns		Name / Description
	NOME	PRIMARY

1.2. Table: foto


Columns

Name		Data type	Description / Attributes
	 IDFOTO	smallint(5, 0)	Identity / Auto increment
	EMAIL_UTENTE	varchar(30)	References: utente
	PATHFOTO	varchar(100)	

Links to







Table	Join	Title / Name / Description
 utente	foto.EMAIL_UTENTE = utente.EMAIL	foto_ibfk_1

Unique keys




Columns	Name / Description
 IDFOTO	PRIMARY

1.3. Table: passaggio

Columns

Name		Data type	Description / Attributes
	ID_PASSAGGIO	smallint(5, 0)	Identity / Auto increment
	ID_TAPPA	smallint(5, 0)	Nullable References: tappa
	EMAILP	varchar(30)	Nullable References: utente_premium
	EMAILA	varchar(30)	Nullable References: utente_aziendale
	INDIRIZZO_PARTENZA	varchar(30)	
	INDIRIZZO_ARRIVO	varchar(30)	


Links to

Table	Join	Title / Name / Description
 tappa	passaggio.ID_TAPPA = tappa.ID_TRAGITTO	passaggio_ibfk_1
 utente_aziendale	passaggio.EMAILA = utente_aziendale.EMAILA	passaggio_ibfk_3
 utente_premium	passaggio.EMAILP = utente_premium.EMAILP	passaggio_ibfk_2

Unique keys





Columns	Name / Description
 ID_PASSAGGIO	PRIMARY

Triggers

Name	When	Description
 AggiornaPostiDisponibili	After Insert	
<pre> BEGIN UPDATE TAPPA SET TAPPA.POSTI=TAPPA.POSTI-1 WHERE TAPPA.VIA=new.INDIRIZZO_ARRIVO and tappa.ID_TRAGITTO=new.ID_TAPPA ; END </pre>		

1.4. Table: pdf


Columns

Name		Data type	Description / Attributes
	 IDPDF	smallint(5, 0)	Identity / Auto increment
	NOME_SOCIETA	varchar(30)	References: pubblica
	PATH	varchar(100)	

Links to









Table	Join	Title / Name / Description
 pubblica	pdf.NOME_SOCIETA = pubblica.NOME	pdf_ibfk_1

Unique keys





Columns	Name / Description
 IDPDF	PRIMARY

1.5. Table: prenotazione

Columns

Name		Data type	Description / Attributes
	ID	smallint(5, 0)	Identity / Auto increment
	INIZIO	datetime	Default: CURRENT_TIMESTAMP
	FINE	datetime	Nullable
	NOTE	varchar(300)	Nullable
	AUTO	varchar(10)	References: veicolo
	UTENTE	varchar(30)	References: utente
	INDIRIZZO_PARTENZA	varchar(30)	References: sosta
	INDIRIZZO_ARRIVO	varchar(30)	References: sosta

Links to

Table		Join	Title / Name / Description
	sosta	prenotazione .INDIRIZZO_PARTENZA = sosta.INDIRIZZO	prenotazione_ibfk_1
	sosta	prenotazione .INDIRIZZO_ARRIVO = sosta.INDIRIZZO	prenotazione_ibfk_2
	utente	prenotazione .UTENTE = utente.EMAIL	prenotazione_ibfk_3
	veicolo	prenotazione .AUTO = veicolo.TARGA	prenotazione_ibfk_4

Linked from



Table		Join	Title / Name / Description
	tragitto_prenotazione	prenotazione .ID = tragitto_prenotazione.ID_PREN	tragitto_prenotazione_ibfk_2

Unique keys


Columns		Name / Description
	ID	PRIMARY

1.6. Table: privata

Columns

Name		Data type	Description / Attributes
	 NOME	varchar(30)	



Unique keys

Columns		Name / Description
	NOME	PRIMARY

TRIAL

1.7. Table: pubblica

Columns

Name		Data type	Description / Attributes
	 NOME	varchar(30)	

Linked from









Table	Join	Title / Name / Description
 pdf	pubblica .NOME = pdf.NOME_SOCIETA	pdf_ibfk_1

Unique keys




Columns	Name / Description
 NOME	PRIMARY

1.8. Table: segnalazione

Columns

Name		Data type	Description / Attributes
	 ID	smallint(5, 0)	Identity / Auto increment
	EMAIL	varchar(30)	Nullable References: utente
	SOCIETA	varchar(30)	Nullable References: societa
	DATA	date	
	TITOLO	varchar(20)	
	TESTO	varchar(200)	
	AUTO	varchar(10)	References: veicolo

Links to





Table		Join	Title / Name / Description
	societa	segnalazione.SOCIETA = societa.NOME	segnalazione_ibfk_3
	utente	segnalazione.EMAIL = utente.EMAIL	segnalazione_ibfk_2
	veicolo	segnalazione.AUTO = veicolo.TARGA	segnalazione_ibfk_1

Unique keys

Columns		Name / Description
	ID	PRIMARY

1.9. Table: societa


Columns

Name		Data type	Description / Attributes
		NOME	varchar(30)
		URL	varchar(30)
		TELEFONO	int(10, 0)

Linked from






Table		Join	Title / Name / Description
	segnalazione	societa .NOME = segnalazione.SOCIETA	segnalazione_ibfk_3
	veicolo	societa .NOME = veicolo.SOCIETA	veicolo_ibfk_1

Unique keys



Columns		Name / Description
	NOME	PRIMARY

1.10. Table: sosta

Columns

Name		Data type	Description / Attributes
	 INDIRIZZO	varchar(30)	
	LAT	decimal(10, 6)	
	LNG	decimal(10, 6)	
	RICARICA	varchar(2)	Nullable

Linked from










Table	Join	Title / Name / Description
 prenotazione	sosta .INDIRIZZO = prenotazione.INDIRIZZO_PARTENZA	prenotazione_ibfk_1
 prenotazione	sosta .INDIRIZZO = prenotazione.INDIRIZZO_ARRIVO	prenotazione_ibfk_2

Unique keys


Columns	Name / Description
 INDIRIZZO	PRIMARY

1.11. Table: tappa


Columns

Name		Data type	Description / Attributes
 	ID_TRAGITTO	smallint(5, 0)	References: tragitto
	CITTA	varchar(20)	
 	VIA	varchar(30)	
	LAT	float(10, 6)	
	LNG	float(10, 6)	
	ORARIO_ARRIVO	datetime	
	POSTI	smallint(5, 0)	Nullable

Links to

Table	Join	Title / Name / Description
 tragitto	tappa .ID_TRAGITTO = tragitto.ID	tappa_ibfk_1

Linked from






Table	Join	Title / Name / Description
 passaggio	tappa .ID_TRAGITTO = passaggio.ID_TAPPA	passaggio_ibfk_1

Unique keys

Columns	Name / Description
 ID_TRAGITTO, VIA	PRIMARY

1.12. Table: tragitto



Columns

Name		Data type	Description / Attributes
	ID	smallint(5, 0)	Identity / Auto increment
	EMAILP	varchar(30)	Nullable References: utente_premium
	EMAILA	varchar(30)	Nullable References: utente_aziendale
	KM	smallint(5, 0)	
	TIPO	enum	Nullable

Links to

Table	Join	Title / Name / Description
 utente_aziendale	tragitto.EMAILA = utente_aziendale.EMAILA	tragitto_ibfk_2
 utente_premium	tragitto.EMAILP = utente_premium.EMAILP	tragitto_ibfk_1

Linked from





Table	Join	Title / Name / Description
 tappa	tragitto.ID = tappa.ID_TRAGITTO	tappa_ibfk_1
 tragitto_prenotazione	tragitto.ID = tragitto_prenotazione.ID_TRAG	tragitto_prenotazione_ibfk_1

Unique keys



Columns	Name / Description
 ID	PRIMARY

1.13. Table: tragitto_prenotazione


Columns

Name		Data type	Description / Attributes
	 ID_TRAG	smallint(5, 0)	References: tragitto
	 ID_PREN	smallint(5, 0)	References: prenotazione

Links to








Table		Join	Title / Name / Description
	prenotazione	tragitto_prenotazione.ID_PREN = prenotazione.ID	tragitto_prenotazione_ibfk_2
	tragitto	tragitto_prenotazione.ID_TRAG = tragitto.ID	tragitto_prenotazione_ibfk_1

Unique keys

Columns		Name / Description
	ID_TRAG, ID_PREN	PRIMARY

1.14. Table: utente

Columns

Name		Data type	Description / Attributes
	 EMAIL	varchar(30)	
	PW	varchar(30)	
	NOME	varchar(30)	
	COGNOME	varchar(30)	
	DATANASCITA	date	
	LUOGO	varchar(50)	Nullable

Linked from




Table	Join	Title / Name / Description
 foto	utente .EMAIL = foto.EMAIL_UTENTE	foto_ibfk_1
 prenotazione	utente .EMAIL = prenotazione.UTENTE	prenotazione_ibfk_3
 segnalazione	utente .EMAIL = segnalazione.EMAIL	segnalazione_ibfk_2
 utente_aziendale	utente .EMAIL = utente_aziendale.EMAILA	utente_aziendale_ibfk_1
 valutazione	utente .EMAIL = valutazione.UTENTE	valutazione_ibfk_2

Unique keys


Columns	Name / Description
 EMAIL	PRIMARY

1.15. Table: utente_aziendale



Columns

Name		Data type	Description / Attributes
	 EMAILA	varchar(30)	References: utente
	NOMEAZIENDA	varchar(30)	Nullable

Links to

Table	Join	Title / Name / Description
 utente	utente_aziendale .EMAILA = utente.EMAIL	utente_aziendale_ibfk_1

Linked from



Table	Join	Title / Name / Description
 passaggio	utente_aziendale .EMAILA = passaggio.EMAILA	passaggio_ibfk_3
 tragitto	utente_aziendale .EMAILA = tragitto.EMAILA	tragitto_ibfk_2

Unique keys




Columns	Name / Description
 EMAILA	PRIMARY

1.16. Table: utente_premium


Columns

Name		Data type	Description / Attributes
	 EMAILP	varchar(30)	

Linked from



Table		Join	Title / Name / Description
	passaggio	utente_premium .EMAILP = passaggio.EMAILP	passaggio_ibfk_2
	tragitto	utente_premium .EMAILP = tragitto.EMAILP	tragitto_ibfk_1
	valutazione	utente_premium .EMAILP = valutazione.EMAIL	valutazione_ibfk_1

Unique keys


Columns		Name / Description
	EMAILP	PRIMARY

1.17. Table: utente_semplice

Columns

Name		Data type	Description / Attributes
	 EMAILS	varchar(30)	








Unique keys

Columns		Name / Description
	EMAILS	PRIMARY



TRIAL

1.18. Table: valutazione

Columns

Name		Data type	Description / Attributes
	 ID	smallint(5, 0)	Identity / Auto increment
	EMAIL	varchar(30)	References: utente_premium
	DATA	datetime	Default: CURRENT_TIMESTAMP
	TESTO	varchar(500)	
	VOTO	smallint(5, 0)	
	UTENTE	varchar(30)	References: utente

Links to










Table	Join	Title / Name / Description
 utente	valutazione.UTENTE = utente.EMAIL	valutazione_ibfk_2
 utente_premium	valutazione.EMAIL = utente_premium.EMAILP	valutazione_ibfk_1

Unique keys

Columns		Name / Description
 ID		PRIMARY

1.19. Table: veicolo



Columns

Name		Data type	Description / Attributes
	TARGA	varchar(10)	
	MODELLO	varchar(20)	
	CAPIENZA	smallint(5, 0)	
	DESCRIZIONE	varchar(200)	
	FERIALE	smallint(5, 0)	
	FESTIVO	smallint(5, 0)	
	SOCIETA	varchar(30)	References: societa
	AREA_SOSTA	varchar(30)	
	STATO	enum	Nullable Default: NON IN USO

Links to

Table	Join	Title / Name / Description
 societa	veicolo.SOCIETA = societa.NOME	veicolo_ibfk_1

Linked from

Table	Join	Title / Name / Description
 prenotazione	veicolo.TARGA = prenotazione.AUTO	prenotazione_ibfk_4
 segnalazione	veicolo.TARGA = segnalazione.AUTO	segnalazione_ibfk_1

Unique keys



Columns	Name / Description
 TARGA	PRIMARY

2. Views

2.1. View: media_voto_utente

VIEW

Columns

Name		Data type	Description / Attributes
	UTENTE	varchar(30)	
	MEDIA_VOTO	decimal(9, 4)	Nullable







Script

```
select `epool`.`valutazione`.`UTENTE` AS `UTENTE`,avg(`epool`.`valutazione`.`VOTO`) AS `MEDIA_VOTO` from  
`epool`.`valutazione` group by `epool`.`valutazione`.`UTENTE` order by avg(`epool`.`valutazione`.`VOTO`) desc
```

2.2. View: prenotazione_capienza

VIEW

Columns

	Name	Data type	Description / Attributes
	AUTO	varchar(10)	
	POSTI_DISPONIBILI	int(10, 0)	Default: 0
	IDTRAGITTO	smallint(5, 0)	Default: 0
	GUIDATORE	varchar(30)	
	TAPPA_CITTA	varchar(20)	
	TAPPA_VIA	varchar(30)	









Script

```
select distinct `epool`.`veicolo`.`TARGA` AS `AUTO`, (`epool`.`veicolo`.`CAPIENZA` - 1) AS  
`POSTI_DISPONIBILI`, `epool`.`tragitto`.`ID` AS `IDTRAGITTO`, `epool`.`prenotazione`.`UTENTE` AS  
`GUIDATORE`, `epool`.`tappa`.`CITTA` AS `TAPPA_CITTA`, `epool`.`tappa`.`VIA` AS `TAPPA_VIA` from `epool`.`prenotazione` join  
`epool`.`veicolo` join `epool`.`tragitto_prenotazione` join `epool`.`tragitto` join `epool`.`tappa` where  
((`epool`.`prenotazione`.`AUTO` = `epool`.`veicolo`.`TARGA`) and (`epool`.`prenotazione`.`ID` =  
`epool`.`tragitto_prenotazione`.`ID_PREN`) and (`epool`.`tragitto`.`ID` = `epool`.`tragitto_prenotazione`.`ID_TRAG`) and  
(`epool`.`tappa`.`ID_TRAGITTO` = `epool`.`tragitto`.`ID`))
```


2.3. View: veicoli_disponibili

VIEW

Columns

Name		Data type	Description / Attributes
	TARGA	varchar(10)	
	MODELLO	varchar(20)	
	CAPIENZA	smallint(5, 0)	
	DESCRIZIONE	varchar(200)	
	FERIALE	smallint(5, 0)	
	FESTIVO	smallint(5, 0)	
	SOCIETA	varchar(30)	
	AREA_SOSTA	varchar(30)	

Script

```
select `epool`.`veicolo`.`TARGA` AS `TARGA`,`epool`.`veicolo`.`MODELLO` AS `MODELLO`,`epool`.`veicolo`.`CAPIENZA` AS `CAPIENZA`,`epool`.`veicolo`.`DESCRIZIONE` AS `DESCRIZIONE`,`epool`.`veicolo`.`FERIALE` AS `FERIALE`,`epool`.`veicolo`.`FESTIVO` AS `FESTIVO`,`epool`.`veicolo`.`SOCIETA` AS `SOCIETA`,`epool`.`veicolo`.`AREA_SOSTA` AS `AREA_SOSTA` from `epool`.`veicolo` where (`epool`.`veicolo`.`STATO` = 'NON IN USO')
```

3. Procedures

3.1. Procedure: getFoto

Input/Output

Name		Data type	Description
→@	Emailt	varchar(30)	

Script

```
BEGIN
  SELECT * FROM FOTO WHERE Emailt=FOTO.EMAIL_UTENTE;
END
```

3.2. Procedure: getPDF

Input/Output

Name		Data type	Description
→@	Nome	varchar(30)	

Script

```
BEGIN
  SELECT * FROM PDF WHERE Nome=PDF.NOME_SOCIETA;
END
```

TRIAL

3.3. Procedure: inserisciPassaggio

Input/Output

	Name	Data type	Description
→@	idN	smallint(5, 0)	
→@	EmailzN	varchar(30)	
→@	partenzaN	varchar(30)	
→@	arrivoN	varchar(30)	

Script

```
BEGIN

    IF EXISTS ( SELECT EMAILP
                FROM UTENTE_PREMIUM
                WHERE EMAILP = EmailzN )

    THEN INSERT INTO PASSAGGIO (ID_TAPPA, EMAILP, INDIRIZZO_PARTENZA, INDIRIZZO_ARRIVO)
                                values (idN, EmailzN, partenzaN, arrivoN);

    CALL printf ('INSERIMENTO PREMIUM AVVENUTO');

else
call printf('Cazzo vuoi 1');
END IF;

    IF EXISTS (SELECT EMAILA
                FROM UTENTE_AZIENDALE
                WHERE EMAILA = EmailzN )

    THEN INSERT INTO PASSAGGIO (ID_TAPPA, EMAILA, INDIRIZZO_PARTENZA, INDIRIZZO_ARRIVO)
                                values (idN, EmailzN, partenzaN, arrivoN);

    CALL printf ('INSERIMENTO AZINEDALE AVVENUTO');

    else
    call printf('Cazzo vuoi 2');
END IF;
END
```

3.4. Procedure: InserisciPrenotazione

Input/Output

	Name	Data type	Description
→@	Notet	varchar(300)	
→@	Automobile	varchar(10)	
→@	Emailt	varchar(30)	
→@	IndirizzoPartenzat	varchar(30)	
→@	IndirizzoArrivot	varchar(30)	
→@	result	tinyint(3, 0)	

Script

```
BEGIN
  start transaction;
  SET result = (FALSE);
  INSERT INTO `prenotazione` ( `NOTE`, `AUTO`, `UTENTE`, `INDIRIZZO_PARTENZA`, `INDIRIZZO_ARRIVO`) VALUES ( Notet,
Automobile, Emailt, IndirizzoPartenzat, IndirizzoArrivot);
  SET result = (TRUE);
  commit work;
END
```

3.5. Procedure: InserisciPrenotazioneAziendale

Input/Output

	Name	Data type	Description
→@	Tragitto	smallint(5, 0)	
→@	Notet	varchar(300)	
→@	Automobile	varchar(10)	
→@	Emailt	varchar(30)	
→@	IndirizzoPartenzat	varchar(30)	
→@	IndirizzoArrivot	varchar(30)	
→@	result	tinyint(3, 0)	

Script

```
BEGIN
    start transaction;
    SET result = (FALSE);
    INSERT INTO `prenotazione` ( `NOTE`, `AUTO`, `UTENTE`, `INDIRIZZO_PARTENZA`, `INDIRIZZO_ARRIVO`) VALUES ( Notet,
Automobile, Emailt, IndirizzoPartenzat, IndirizzoArrivot);
    SET result = (SELECT LAST_INSERT_ID());
    INSERT INTO `tragitto_prenotazione` ( `ID_TRAG`, `ID_PREN`) VALUES ( Tragitto, result);
    SET result = (TRUE);
    commit work;
END
```

3.6. Procedure: InserisciSegnalazione

Input/Output

	Name	Data type	Description
→@	Emailt	varchar(30)	
→@	SocietaAutomobile	varchar(30)	
→@	DataSegnalazione	date	
→@	TitoloSegnalazione	varchar(20)	
→@	TestoSegnalazione	varchar(200)	
→@	Automobile	varchar(10)	
→@	result	tinyint(3, 0)	

Script

```
BEGIN
  start transaction;
  SET result = (FALSE);
  INSERT INTO `segnalazione` (`EMAIL`, `SOCIETA`, `DATA`, `TITOLO`, `TESTO`, `AUTO`) VALUES (Emailt, SocietaAutomobile,
DataSegnalazione, TitoloSegnalazione, TestoSegnalazione, Automobile);
  SET result = (TRUE);
  commit work;
END
```

3.7. Procedure: InserisciTappa

Input/Output

	Name	Data type	Description
→@	idN	smallint(5, 0)	
→@	cittaN	varchar(20)	
→@	viaN	varchar(30)	
→@	orarioN	datetime	

Script

```
BEGIN
    DECLARE x SMALLINT;
    SET x = (SELECT CAPIENZA FROM veicolo,tragitto_prenotazione, prenotazione
    WHERE tragitto_prenotazione.ID_TRAG=idN AND
            tragitto_prenotazione.ID_PREN=prenotazione.ID AND
            veicolo.TARGA=prenotazione.AUTO
    limit 1);
    IF EXISTS (SELECT ID
                FROM TRAGITTO
                WHERE ID = idN)
    THEN INSERT INTO TAPPA (ID_TRAGITTO, CITTA, VIA, ORARIO_ARRIVO, POSTI)
        VALUES (idN, cittaN, viaN, orarioN, X-1 );

    ELSE
        CALL printf ('[ERRORE], NON ESISTE NESSUN TRAGITTO CON QUESTO ID');
    END IF;
END
```


3.8. Procedure: inserisciTragitto

Input/Output

	Name	Data type	Description
→@	EmailN	varchar(30)	
→@	km	smallint(5, 0)	
→@	tipe	varchar(30)	
↵@	res	smallint(5, 0)	

Script

```
BEGIN
  IF EXISTS ( SELECT EMAILP
              FROM UTENTE_PREMIUM
              WHERE EMAILP = EmailN )

  THEN INSERT INTO TRAGITTO (EMAILP, KM, TIPO) values (EmailN, km, tipe);
  SET res = (SELECT LAST_INSERT_ID());
  ELSE
    CALL printf('QUALCOSA PREMIUM è ANDATO STORTO');
  END IF;

  IF EXISTS (SELECT EMAILA
            FROM UTENTE_AZIENDALE
            WHERE EMAILA = EmailN )

  THEN INSERT INTO TRAGITTO (EMAILA, KM, TIPO) values (EmailN, km, tipe);
  CALL printf ('INSERIMENTO AZIENDALE AVVENUTO');
  SET res = (SELECT LAST_INSERT_ID());
  ELSE
    CALL printf('QUALCOSA AZIENDALE è ANDATO STORTO');
  END IF;
END
```

3.9. Procedure: InserisciValutazione

Input/Output

	Name	Data type	Description
→@	Emailt	varchar(30)	
→@	TestoValutazione	varchar(500)	
→@	VotoValutazione	smallint(5, 0)	
→@	UtenteV	varchar(30)	
→@	result	tinyint(3, 0)	

Script

```
BEGIN
    start transaction;
    SET result = (FALSE);
    INSERT INTO `valutazione` (`EMAIL`, `DATA`, `TESTO`, `VOTO`, `UTENTE`) VALUES (Emailt, CURRENT_DATE, TestoValutazione,
VotoValutazione, UtenteV);
    SET result = (TRUE);
    commit work;
END
```

3.10. Procedure: InsertFoto

Input/Output

	Name	Data type	Description
→@	Emailt	varchar(30)	
→@	Path	varchar(100)	
↔@	result	tinyint(3, 0)	

Script

BEGIN

```
START TRANSACTION;  
SET result = (FALSE);  
INSERT INTO Foto (`EMAIL_UTENTE`, `PATHFOTO`) VALUES (Emailt, Path);  
COMMIT WORK;  
SET result = (TRUE);
```

END

3.11. Procedure: InsertPdf

Input/Output

	Name	Data type	Description
→@	Nome	varchar(30)	
→@	Pathpdf	varchar(100)	
↩@	result	tinyint(3, 0)	

Script

```
BEGIN

  START TRANSACTION;
  SET result = (FALSE);
  INSERT INTO PDF (`NOME_SOCIETA`, `PATH`) VALUES (Nome, Pathpdf);
  COMMIT WORK;
  SET result = (TRUE);

END
```

3.12. Procedure: Login

Input/Output

	Name	Data type	Description
→@	Email	varchar(30)	
→@	pasw	varchar(30)	
↔@	result	tinyint(3, 0)	

Script

```
BEGIN DECLARE statoUtente int; IF EXISTS (
  SELECT
    *
  FROM
    UTENTE
  WHERE
    UTENTE.EMAIL = Email
    AND UTENTE.PW = pasw
) THEN
SET
  result = (TRUE);
ELSE
SET
  result = (FALSE); END IF;
END
```

3.13. Procedure: LoginAziendale

Input/Output

	Name	Data type	Description
→@	Email	varchar(30)	
→@	pasw	varchar(30)	
↔@	result	tinyint(3, 0)	

Script

```
BEGIN
  IF EXISTS(SELECT *
            FROM UTENTE
            WHERE UTENTE.EMAIL = Email AND UTENTE.PW=pasw) THEN
    IF EXISTS(SELECT *
              FROM UTENTE_AZIENDALE
              WHERE UTENTE_AZIENDALE.EMAILa = Email) THEN
      SET result = (TRUE);
    ELSE
      SET result = (FALSE);
    END IF;
  ELSE
    SET result = (FALSE);
  END IF;
END
```

3.14. Procedure: LoginType

Input/Output

	Name	Data type	Description
→@	Email	varchar(30)	
→@	result	tinyint(3, 0)	

Script

```
BEGIN
  IF EXISTS (SELECT *
             FROM UTENTE_PREMIUM
             WHERE UTENTE_PREMIUM.EMAILP = Email) THEN
    SET result = 2;
  ELSE
    SET result = 1;
  END IF;
END
```

3.15. Procedure: PrenotazioneP

Input/Output

	Name	Data type	Description
→@	NoteQ	varchar(300)	
→@	AutoQ	varchar(10)	
→@	UtenteQ	varchar(30)	
→@	Partenza	varchar(30)	
→@	Arrivo	varchar(30)	
↩@	result	tinyint(3, 0)	

Script

```
BEGIN
  start transaction;
  SET result = (FALSE);
  INSERT INTO PRENOTAZIONE (NOTE, AUTO, UTENTE, INDIRIZZO_PARTENZA, INDIRIZZO_ARRIVO)
  VALUES ( NoteQ, AutoQ, UtenteQ, Partenza, Arrivo);
  commit work;
  SET result = (TRUE);
END
```


3.16. Procedure: printf

Input/Output

Name		Data type	Description
→@	mytext	text	

Script

```
BEGIN
  select mytext as ``;
END
```

TRIAL

3.17. Procedure: RegistrazioneAziendale

Input/Output

Name		Data type	Description
✉@	Emailt	varchar(30)	
✉@	pasw	varchar(30)	
✉@	nome	varchar(30)	
✉@	cognome	varchar(30)	
✉@	datanascita	date	
✉@	luogo	varchar(50)	
✉@	nomeazienda	varchar(30)	
✉@	result	tinyint(3, 0)	

Script

```
BEGIN
    start transaction;
    SET result = (FALSE);
    IF NOT EXISTS ( SELECT EMAIL
                    FROM UTENTE
                    WHERE EMAIL = Emailt)
        THEN IF EXISTS (SELECT *
                        FROM AZIENDA
                        WHERE NOMEAZIENDA = nomeazienda)
            THEN
                INSERT INTO UTENTE (EMAIL, PW, NOME, COGNOME, DATANASCITA, LUOGO) VALUES (Emailt, pasw, nome, cognome,
                datanascita, luogo);
                INSERT INTO UTENTE_AZIENDALE (EMAILA, NOMEAZIENDA) VALUES (Emailt, nomeazienda);
                SET result = (TRUE);
                commit work;
            ELSE
                CALL printf ('[ERRORE] AZIENDA NON PRESENTE NEL SISTEMA');
                rollback;
            END IF;
        ELSE
            CALL printf ('[ERRORE] UTENTE GIA'' REGISTRATO');
            rollback;
        END IF;
END
```

3.18. Procedure: RegistrazioneUtente

Input/Output

	Name	Data type	Description
→@	EmailN	varchar(30)	
→@	pasw	varchar(30)	
→@	nome	varchar(30)	
→@	cognome	varchar(30)	
→@	datanascita	date	
→@	luogo	varchar(50)	

Script

```
BEGIN
  start transaction;
  IF NOT EXISTS ( SELECT EMAIL
                  FROM UTENTE
                  WHERE EMAIL = EmailN)
  THEN
    INSERT INTO UTENTE (EMAIL, PW, NOME, COGNOME, DATANASCITA, LUOGO) VALUES (EmailN, pasw, nome, cognome, datanascita,
luogo);
    commit work;
  ELSE
    CALL printf ('[ERRORE] UTENTE GIA'' REGISTRATO');
    rollback;
  END IF;
END
```